# Moving in temporal graphs with very sparse random availability of edges

Paul G. Spirakis[a,b], Eleni Ch. Akrida[c]

[a] *Computer Technology Institute & Press Diophantus (CTI), Patras, Greece*
[b] *Department of Computer Science, University of Liverpool, UK*
[c] *Department of Mathematics, University of Patras, Greece*
*Email:* spirakis@cti.gr, akridel@master.math.upatras.gr

## Abstract

In this work we consider temporal graphs, i.e. graphs, each edge of which is assigned a set of discrete time-labels drawn from a set of integers. The labels of an edge indicate the discrete moments in time at which the edge is available. We also consider temporal paths in a temporal graph, i.e. paths whose edges are assigned a strictly increasing sequence of labels. Furthermore, we assume the uniform case (UNI-CASE), in which every edge of a graph is assigned exactly one time label from a set of integers and the time labels assigned to the edges of the graph are chosen randomly and independently, with the selection following the uniform distribution. We call uniform random temporal graphs the graphs that satisfy the UNI-CASE. We begin by deriving the expected number of temporal paths of a given length in the uniform random temporal clique. We define the term temporal distance of two vertices, which is the arrival time, i.e. the time-label of the last edge, of the temporal path that connects those vertices, which has the smallest arrival time amongst all temporal paths that connect those vertices. We then propose two statistical properties of temporal graphs. One is the maximum expected temporal distance which is, as the term indicates, the maximum of all expected temporal distances in the graph. The other one is the temporal diameter which, loosely speaking, is the expectation of the maximum temporal distance in the graph. Since uniform random temporal graphs, except for the clique, have at least a pair of vertices whose temporal distance is infinity, we assume the existence of a *slow* way to go directly from any vertex to any other vertex in order for the above measures to have a finite value. We derive the maximum expected temporal distance of a uniform random temporal star graph as well as an $O(\sqrt{n}\log^2 n)$ upper bound, and a greedy algorithm which computes in polynomial time the path that achieves it, on both the maximum expected temporal distance and the temporal diameter of the *normalized* version of the uniform random temporal clique, in which the largest time-label available equals the number of vertices. Finally, we provide an algorithm that solves an optimization problem on a specific type of temporal (multi)graphs of two vertices.

*Keywords:* Temporal graphs; Probabilistic analysis of algorithms; The bridges' optimization problem

## 1. Introduction

A temporal graph (or otherwise called temporal network) is, loosely speaking, a graph that changes with time. This concept incorporates a variety of both modern and traditional networks such as information and communication networks, social networks, transportation networks, and several physical systems. The presence of dynamicity in modern communication networks, i.e. in mobile ad hoc, sensor, peer-to-peer, and delay-tolerant networks, is often very strong. We can also find that kind of dynamicity in social networks, where the topology usually represents the social connections between a group of individuals. Those connections change as the social relationships between the individuals or even the individuals themselves change. Temporal graphs can also be associated with transportation networks. In a transportation network, there is usually some fixed network of routes and a set of transportation units moving over these routes. In such networks, the dynamicity refers to the change of positions of the transportation units in the network as time passes. Concerning physical systems, dynymicity may be present in systems of interacting particles.

In this work, embarking from the foundational work of Kempe et al. [KKK00], we consider the time to be discrete, that is, we consider networks in which changes can only occur at discrete moments in time, e.g. days or hours. This choice not only gives to the resulting models a purely combinatorial flavor but also naturally abstracts many real systems. In particular, we consider those networks that can be described via an underlying graph $G$ and a labeling $L$ assigning a set of discrete labels to each edge of $G$. This is a generalization of the single-label-per-edge model used in [KKK00], as we allow many time-labels to appear on an edge, although in this work we mainly focus on single-labeled temporal graphs. These labels are drawn from the natural numbers and indicate the discrete moments in time at which the corresponding connection is available, i.e. the corresponding edge exists in the graph. For example, in a communication network, the availability of a connection at some time $t$ may indicate that a communication protocol is allowed to transmit a data packet over that connection at time $t$. A temporal path (or journey) in a temporal graph is a path, on the edges of which we can find strictly ascending time labels. The number of edges on the latter is called length of the temporal path. This, for a communication network,

would mean that it is possible to transmit a data packet along the network nodes that belong to such a path from the first node in order to the last one, as time progresses. The time label on the last edge of a temporal path is called its arrival time and, in the above example of a connection network, it would indicate the time at which the transmitted data packet would arrive at the last node of the path.

In this work, we initiate the study of temporal graphs from a probabilistic and statistical viewpoint. In particular, we consider the case in which every edge of a graph is assigned exactly one time label from a set $L_0 = \{1, 2, \ldots, a\}$ of integers. The time labels assigned to the edges of the graph are chosen *randomly* and *independently* from one another from the set $L_0$ and the probability that an edge is assigned a time label $i \in L_0$ is equal to $\frac{1}{a}$, for every $i \in L_0$. We use the term *UNI-CASE* for the above described case and for any graph that satisfies UNI-CASE's properties we use the term *Uniform Random Temporal Graph*. We focus on examining three statistical properties of such graphs. The first one, called *expected number of temporal paths of a given length*, is the number of temporal paths, of a given length, that we expect to have in a graph, given that every edge is assigned a label satisfying UNI-CASE. The second one, called *the Maximum Expected Temporal Distance*, is the maximum of all temporal distances in the graph. By temporal distance of two vertices we denote the arrival time of the temporal path that connects those vertices, which has the smallest arrival time amongst all temporal paths that connect those vertices. The last property that we examine is called *the Temporal Diameter* of a uniform random temporal graph. Loosely speaking, it is the expected value of the maximum temporal distance in the graph, which of course is in correspondence with the diameter of a graph, as we know it up to now.

The motivation of the definitions we initiate and the work we carry out here comes from the natural question on how fast we can visit a particular destination, i.e. arrive at a particular network node, starting from a given point of origin, i.e. another network node, when the connection between a pair of nodes only exists at one moment in time.

### *1.1. Related work*

**Labeled Graphs.** Labeled graphs are becoming an increasingly useful family of Mathematical Models for a broad range of applications both in Computer Science and in Mathematics, e.g. in Graph Coloring[MR02]. In our work, labels correspond to time moments of availability and the properties of

labeled graphs that we study are naturally *temporal properties*. However, we can note that any property of a graph that is assigned labels from a discrete set of labels can correspond to some temporal property. Take for example a proper edge-coloring in a graph, i.e. a coloring of the graph's edges in which no two adjacent edges have the same color. This corresponds to a temporal graph in which no two adjacent edges have the same time label, that is no two adjacent edges exist at the same time.

**Single-labeled and multi-labeled Temporal Graphs.** The model of temporal graphs that we consider in this work has a direct relation with the single-labeled model studied in [KKK00] as well as the multi-labeled model studied in [MMCS13]. The main results of [KKK00] and [MMCS13] have to do mainly with connectivity properties and/or cost minimization parameters for temporal network design. In this work we study temporal graphs from a statistical view and mainly focus on how fast we expect to arrive at a target vertex in a temporal graph. In [KKK00], a temporal path is considered to be a path with non-decreasing labels on its edges. In this work, we follow the assumption of [MMCS13] and consider a temporal path to be a path with strictly increasing labels. This choice is also motivated by recent work on dynamic communication systems, in which if it takes one time unit for the transmition of a data packet over a link, then a packet can only be transmitted over paths with strictly increasing labels.

**Continuous Availabilities (Intervals).** Some authors have assumed the availability of an edge for a whole time-interval $[t_1, t_2]$ or multiple such time-intervals. Although this is a clearly natural assumption, in this work we focus on the availability of edges at discrete moments and we design and develop techniques which are quite different from those needed in the continuous case.

### *1.2. Roadmap and contribution*

In Section 2, we formally define the model of temporal graphs under consideration and provide all further necessary basic definitions. In Section 3, we make some general remarks on the expected number of temporal paths in any graph and proceed to the study of the expected number of temporal paths of a given length in the uniform random temporal clique of $n$ vertices, $K_n$. For this matter, we distinguish two cases. In Section 3.1, we study the first case, where we set the largest label available for assignment to be $a = n-1$ and we search for the expected number of temporal paths of length $k = n - 1$. In Section 3.2, we study the second case, where we loosen the parameters $a$ and $k$ and we look at the expected number of temporal paths

of length $k < a$, when the largest label available for assignment is $a = n - 1$. In Section 4, we formally define the maximum expected temporal distance of a uniform random temporal graph and we make some preliminary notations. In Section 4.1, we look at some known graphs' maximum expected temporal distance. In particular, in Section 4.1.1, we study the case of the uniform random temporal star graph and we provide its exact maximum expected temporal distance. In Section 4.1.2, we study the case of the uniform random temporal clique, focusing on its normalized version, where the largest label, $a$, available for assignment is equal to the number of vertices, $n$. We also give a simple *(greedy)* algorithm which can, with high probability, find a temporal path with small expected arrival time from a given source to a given target vertex in the normalized uniform random temporal clique. In Section 5, we formally define the temporal diameter of a uniform random temporal graph and provide an inequality relation between the latter and the maximum expected temporal distance as well as the relevant proof. Furthermore, we provide an upper bound for both the temporal diameter and the maximum expected temporal distance of the nomalized uniform random temporal clique. In Section 6, we study an optimization problem on a specific type of temporal (multi)graphs of two vertices. We prove that the problem can by polynomially solved and provide an algorithm that gives the solution, along with the proof of its correctness. Finally, in Section 7 we conclude and give further research horizons opened through our work.

## 2. Preliminaries

**Definition 1.** *A temporal graph is an ordered triplet $G = \{V, E, L\}$, where:*

- *$V$ stands for a nonempty finite set (called set of vertices)*
- *$E$ stands for a set of $m$ elements, each of which is a 2-element subset of $V$ (called set of edges), and*
- *$L = \{L_e, \forall e \in E\} = \{L_{e_1}, L_{e_2}, \ldots, L_{e_m}\}$, is a set of $m$ elements, $L_{e_i}$, $1 \leq i \leq m$, each of which is a set of positive integers mapped to the edge $e_i \in E$ (called assignment of time labels or simply assignment)*

We also denote the temporal graph $G = \{V, E, L\}$ by $G'(L)$ or $(G', L)$, where $G' = \{V, E\}$ is the graph, on the edges of which we assign the time labels, and $L = \{L_e,\ e \in E(G')\}$ is the assignment.

The values assigned to each edge of the graph are called time labels of the edge and indicate the times at which we can cross it (from one end to the other).

*2.1. Further Definitions*

We can now talk about temporal edges (or time edges) that are considered to be triplets $(u, v, l)$, where $u, v$ are the ends of an edge in the temporal graph and $l \in L_{\{u,v\}}$ is a time label of this edge. That is, if an edge $e = \{u, v\}$ has more than one time labels, e.g. has a set of three time labels, $L_e = \{l_1, l_2, l_3\}$, then this edge has three corresponding time edges, $(u, v, l_1)$, $(u, v, l_2)$ and $(u, v, l_3)$.

**Definition 2.** *A journey $j$ from a vertex $u$ to a vertex $v$ ($(u, v)$- journey) is a sequence of time edges $(u, u_1, l_1)$, $(u_1, u_2, l_2), \ldots, (u_{k-1}, v, l_k)$, such that $l_i < l_{i+1}$, for each $1 \leq i \leq k - 1$.*
*We call the last time label of journey $j$, $l_k$,* a*rrival time of the journey.*

**Definition 3.** *A $(u, v)$-journey $j$ in a temporal graph is called* f*oremost journey if its arrival time is the minimum arrival time of all $(u, v)$-journeys' arrival times, under the labels assigned on the graph's edges.*

Now, consider any temporal graph $G = \{V, E, L\}$. Let every edge receive exactly one time label, chosen randomly, independently of one another from a set $L_0 = \{1, 2, \ldots, a\}$, where $a \in \mathbb{N}$, with the probability of an edge label to be $i$, $\forall i \in L_0$, equal to $\frac{1}{a}$. (**UNI-CASE)**

**Definition 4.** *A temporal graph that satisfies UNI-CASE is called Uniform Random Temporal Graph (U-RTG).*

In the special case, where the largest label, $a$, that can be assigned to the edges of a graph is equal to the number of its vertices, the graph is called *Normalized Uniform Random Temporal Graph (Normalized U-RTG).*

**Note.** There could be prospective study of cases in which each edge of a graph may receive several time labels, selected randomly and independently of one another from the set $L_0 = \{1, 2, \ldots, a\}$, where $a \in \mathbb{N}$, with the selection following a distribution F. (**F-CASE)**
In such cases, the graphs under consideration would be called *F-Random Temporal Graphs (F-RTG)* respectively.

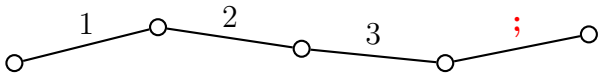
In the following sections, we will look for the expected number of journeys of length k in some well-known graphs that satisfy UNI-CASE. For the sake

of brevity, we often call such journeys *"k edges temporal paths"*. We also study the Expected (or Temporal) Diameter and the Maximum Temporal Distance of a graph, as defined in the following paragraphs.

## 3. Expected number of temporal paths

In this section we will search for the expected number of $k$ edges temporal paths in a clique of n vertices, $K_n$, that satisfies UNI-CASE.

It is obvious that for there to exist a temporal path of length k in **any** graph, the number of edges, k, has to be at most equal to the maximum label of the set $L_0$, $a$, that can be assigned to the various edges. Otherwise, it is impossible for a $k$ edges temporal path to exist (see Figure 1).




$L_0 = \{1, 2, 3(= a)\}$

$k = 4$

Figure 1: There is no temporal path, when $k > a$

### *3.1. Special case:* $G = K_n,\ k = n - 1,\ a = n - 1$

Initially, we focus our interest in the case of the clique (complete graph) of n vertices, $K_n$, that satisfies UNI-CASE with $a = n - 1$ (i.e. with $L_0 = \{1, 2, \ldots, n - 1\}$), in which we seek the expected number of $n - 1$ edges temporal paths.

Obviously, there can only be one assignment of labels of $L_0$ on the $k = n - 1$ edges of any path starting from a random initial vertice $v_0 \in V(K_n)$ in the clique $K_n$ such, that we can find a journey on the edges of this path. This assignment gives label 1 on the $1^{st}$ edge, label 2 on the $2^{nd}$ edge, ... , label $n - 1$ on the $(n - 1)^{th}$ edge.

Each edge can receive exactly one label from a set of $n-1$ labels. Therefore, the total number of assignments that can be made on these $n-1$ edges

is:

$$\#assignments = (n-1)^{n-1}$$

Consequently, given a path of $n-1$ edges starting from $v_0$, the probability for there to exist the corresponding temporal path (i.e. the one arising on the simple path after the assignment of the time labels) is:

$$P(temporal\_path\_of\_length\_n-1\_starting\_from\_v_0) = \frac{1}{(n-1)^{n-1}}$$

The number of paths of length $n-1$, starting from $v_0$ in the clique $K_n$ is equal to the number of permutations of the $n-1$ vertices remaining (i.e. except the start $v_0$) to construct such a path. That is, the number of paths of length $n-1$ that start from $v_0$ in the clique $K_n$ is:

$$(n-1)!$$

Therefore, since the clique $K_n$ has $n$ vertices, and due to the linearity of expectation, the expected number of temporal paths of length $k = n-1$ in the clique $K_n$ is:

$$E(\#temporal\_paths\_of\_length\_n-1) = n \cdot (n-1)! \cdot \frac{1}{(n-1)^{n-1}} = \frac{n!}{(n-1)^{n-1}}$$

*Comments.* Let us observe that when $n$ is too large ($n \to +\infty$), then, by Stirling's formula, we result in the following:

$$\begin{aligned} E(\#temporal\_paths\_of\_length\_n-1) &= \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n}{(n-1)^{n-1}} \\ &= \frac{\sqrt{2\pi n} n^n}{e^n (n-1)^{n-1}} \xrightarrow[n\to+\infty]{} 0 \end{aligned}$$

Of course, this is more or less obvious when we consider the fact that it is difficult to find $n-1$ edges temporal paths in the clique of $n$ vertices when $n$ is too large. This is because in order to have a temporal path of such length, the (so many) time labels should be assigned on the edges so that they maintain the desired strictly increasing sequence, something that is increasingly less likely to happen as $n$ increases.

*3.2. Special case:* $G = K_n,\ k < a,\ a \geq n$

Now let's see what happens in the case of the clique $K_n$, that satisfies UNI-CASE, when we look at the expected number of temporal paths of length $k < a$ and the maximum label that can be assigned to any edge of the clique is $a \geq n$.

Starting from a vertex $v_0 \in V(K_n)$ and along the path of k edges, we can construct, as explained in Figure 2, a number of assignments equal to:

$$\#assignments = a^k$$

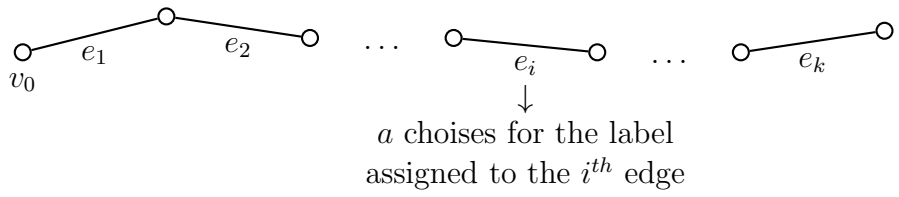



Figure 2: Number of assignments on a path of length $k$, when $k < a$

The number of assignments that can be made on the $k$ edges, where the time labels assigned are distinct (different from each other) is:

$$\#distinct\_time\_labels\_assignments = a \cdot (a-1) \cdot \ldots \cdot (a-k+1) = \frac{a!}{(a-k)!}$$

We will now calculate the number of paths of length $k$ that can be starting from $v_0 \in V(K_n)$. We have $n-1$ options for how to select $v_1$, the vertex following $v_0$ on the path, $n-2$ options for how to select $v_2$, the vertex following $v_1$ on the path, etc., and finally $n-k$ options for how to select $v_k$, the last vertex on the path.

Therefore, the number of paths of length $k$ that can be starting from $v_0 \in V(K_n)$ is:

$$\#paths\_of\_length\_k\_starting\_from\_v_0 = (n-1)\cdot(n-2)\cdot\ldots\cdot(n-k) = \frac{(n-1)!}{(n-k-1)!}$$

We call $A$ the event that "we have the *right* labels" assignment on the $k$ edges of any path of length $k$ starting from $v_0$".

That is, if $l_1, l_2, \ldots, l_k$ are the time labels assigned to the $1^{st}$, the $2^{nd}$, ..., the $k^{th}$ edge of the path, respectively, with $l_i \in L_0 = \{1, 2, \ldots, a\}$, $\forall i = 1, 2, \ldots, k$, $A$ is the event that:

$$l_1 < l_2 < \ldots < l_k$$

We call $\phi$ the probability that $A$ occurs. That is:

$$\phi = P(A) = P(l_1 < l_2 < \ldots < l_k)$$

Let us note that the number of assignments of $k$ labels, $l_{a_i},\ i = 1, \ldots, k$, such that

$$l_{a_1} < l_{a_2} < \ldots < l_{a_k}$$

is $k!$ and each one has a probability equal to $P(A)$ to happen.
Therefore, if we consider $B$ to be the event that "*at least* two of the labels assigned on the $k$ edges of the path are equal", then the following applies:

$$k! \cdot P(A) + P(B) = 1 \Leftrightarrow$$

$$k! \cdot \phi + 1 - P(\urcorner B) = 1 \tag{1}$$

The probability that the event $\urcorner B$ occurs, that is there are no two equal labels assigned on the $k$ edges of the path, is:

$$P(\urcorner B) = \frac{\#distinct\_time\_labels\_assignments}{\#assignments} =$$

$$= \frac{\frac{a!}{(a-k)!}}{a^k}$$

$$= \frac{a!}{a^k \cdot (a-k)!}$$

Consequently, the relation (1) becomes:

$$k! \cdot \phi + 1 - \frac{a!}{a^k \cdot (a-k)!} = 1 \Leftrightarrow$$

$$\Leftrightarrow \phi = \frac{a!}{k! \cdot a^k \cdot (a-k)!}$$

Let us recall that $\phi$ is the probability to have a *proper* assignment on the $k$ edges of any path of length $k$ starting from any vertice $v_0$ of the clique $K_n$.

Also, recall that the number of paths of length $k$ that can be starting from any vertice $v_0$ of the clique $K_n$ is $\frac{(n-1)!}{(n-k-1)!}$.
Therefore, the expected number of paths of length $k$ that start from a random vertex $v_0$ and on which there are labels assigned so that there exists a temporal path on them, is:

$$E(\#temporal\_paths\_of\_length\_k\_starting\_from\_v_0) = \frac{(n-1)!}{(n-k-1)!} \cdot \phi$$

Eventually, since the clique $K_n$ has a number of $n$ vertices, the expected number of paths of length $k$, on which labels are assigned in a way that there exists a temporal path on them, is:

$$\begin{aligned} E(\#temporal\_paths\_of\_length\_k) &= n \cdot \frac{(n-1)!}{(n-k-1)!} \cdot \phi \\ &= \frac{n \cdot (n-1)!}{(n-k-1)!} \cdot \frac{a!}{k! \cdot a^k \cdot (a-k)!} \\ &= \frac{n! \cdot a!}{(n-k-1)! \cdot k! \cdot a^k \cdot (a-k)!} \end{aligned}$$

*Comments.* Let us observe that the probability $\phi$ is:

$$\phi = \frac{1}{k!} \cdot \frac{\overbrace{a(a-1)\dots(a-k+1)}^{\text{k factors}}}{\underbrace{a \cdot \dots \cdot a}_{\text{k factors}}}$$

and so, if $a$ is very large in comparison with $k$, then we have $\phi \approx \frac{1}{k!}$.
Hence, if $a$ is far larger than $k$, then the expected number of temporal paths of length $k$ in the clique $K_n$, is:

$$E(\#temporal\_paths\_of\_length\_k) \approx \frac{n!}{k!(n-k-1)!} = \frac{n \cdot (n-1) \cdot \dots \cdot (n-k)}{k!}$$

## 4. The Maximum Expected Temporal Distance

In this section, we will define and study a new concept, that of *the maximum expected temporal distance* of a U-RTG.
Henceforth, we make the following assumption. For every pair of vertices in any U-RTG, there exists a ***slow*** journey that connects them, whose arrival time is a fixed, for each graph, number $n' \in \mathbb{N}$, where $n'$ is greater than the expected value of any edge's label, $l$. That is $n' \geq E(l)$.

**Definition 5.** *Consider an instance $G(L)$ of a U-RTG. Given two vertices $s, t \in V\big(G(L)\big)$, we define:*

- $\delta'(s,t) = a(j)$, *where $j$ is a foremost $(s,t)-$journey, to be called* ***distributional temporal distance*** *from source vertex $s$ to target vertex $t$ under the assignment $L$. If there exists no $(s,t)-$journey in $G$, then $\delta'(s,t) \to \infty$*
- $\delta(s,t) = min\{\delta'(s,t), n'\}$ *to be called* ***temporal distance*** *from source vertex $s$ to target vertex $t$ under the assignment $L$, and*
- $MD = max_{s,t \in V(G)} E\big(\delta(s,t)\big)$ *to be called* ***Maximum Expected Temporal Distance*** *of $G$*

*Remark.* If the $U - RTG$ is a path itself, then its maximum expected temporal distance is obviously $n'$.

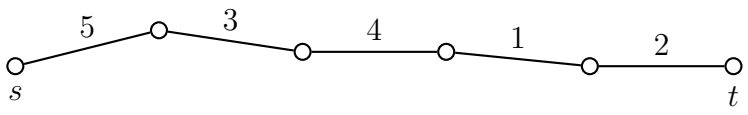



Figure 3: MD of a $U - RTG$, which is a path itself, equals $n'$.

This can be easily understood if we consider that for any two vertices $u$ and $v$ in the path, if there exists a $(u,v)$-journey, then the time labels assigned to its edges form a strictly increasing sequence and thus there is no $(v,u)$-journey in it, apart from the *slow* journey which we assume that exists. Therefore, $\delta'(v,u) \to +\infty$ and $\delta(v,u) = min\{\delta'(v,u), n'\} = n'$. (see Figure 4).

*4.1. Known graphs' maximum expected temporal distance*

Next, we study the maximum expected temporal distance of two known graphs, the star graph of $n$ vertices, which we denote by $G_{star}$ (see Figure 5) and the clique of $n$ vertices, $K_n$ (see Figure 6).

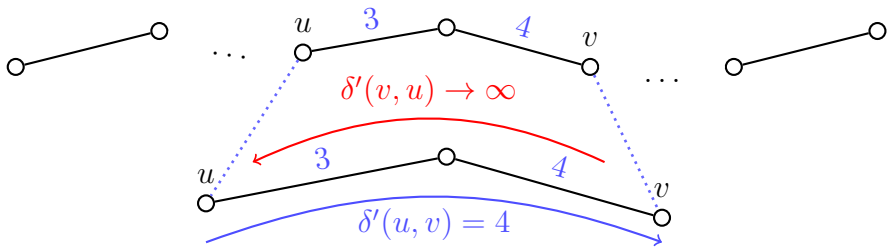



Figure 4: Example of temporal distance, from source vertex to target vertex, equal to $n'$.

*4.1.1. Case: $G = G_{star}$*

It is easy to understand that, even if the temporal star graph does not satisfy UNI-CASE, but satisfies any F-CASE, as defined in Section 2.1, it is:

$$max_{s,t \in V(G_{star})} E_F\big(\delta(s,t)\big) \geq 2, \text{ for any distribution } F$$

We will calculate the exact maximum expected temporal distance, $MD$, of a uniform random temporal star graph. It is:

$$\begin{aligned} MD(G_{star}) &= max_{s,t \in V(G_{star})} E\big(\delta(s,t)\big) \\ &= E\big(\delta(s,t)\big), \text{ for any two vertices } s,t \in V(G_{star}) \\ &= E(l_2 |\ l_2 > l_1) \cdot P(l_2 > l_1) + n' \cdot P(l_2 \leq l_1) \end{aligned} \tag{2}$$

We calculate the expected value of label $l_2$, given that $l_2 > l_1$, that is $E(l_2 |\ l_2 > l_1)$:

$$\begin{aligned} E(l_2 |\ l_2 > l_1) &= \sum_{i=1}^{a} E(l_2 |\ l_2 > i) \cdot P(l_1 = i) \\ &= \sum_{i=1}^{a} \Big( \sum_{i'=i}^{a} \big(P(l_2 = i'+1) \cdot (i'+1)\big) \Big) \cdot P(l_1 = i) \\ &= \sum_{i=1}^{a} \Big( \sum_{i'=i}^{a} (i'+1) \cdot \frac{1}{a} \Big) \cdot \frac{1}{a} \\ &= \frac{1}{a^2} \cdot \sum_{i=1}^{a} \sum_{i'=i}^{a} (i'+1) \end{aligned}$$

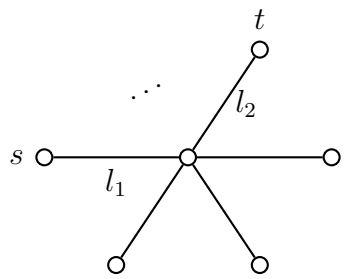



Figure 5: A star graph

$$
\begin{aligned}
&= \frac{1}{a^2} \cdot \Big( \sum_{i'=1}^{a} (i'+1) + \sum_{i'=2}^{a} (i'+1) + \ldots + \sum_{i'=a}^{a} (i'+1) \Big) \\
&= \frac{1}{a^2} \cdot \Big( \big(2+3+\ldots+(a+1)\big) + \big(3+4+\ldots+(a+1)\big) + \ldots + \big((a+1)\big) \Big) \\
&= \frac{1}{a^2} \cdot \Big( 1\cdot 2 + 2\cdot 3 + 3\cdot 4 + 4\cdot 5 + \ldots + a\cdot(a+1) \Big) \\
&= \frac{1}{a^2} \cdot \sum_{i=1}^{a} \Big( i\cdot(i+1) \Big) \\
&= \frac{1}{a^2} \cdot \sum_{i=1}^{a} \Big( i^2 + i \Big) \\
&= \frac{1}{a^2} \cdot \sum_{i=1}^{a} i^2 + \sum_{i=1}^{a} i \\
&= \frac{1}{a^2} \cdot \Big( \frac{a\cdot(a+1)\cdot(2a+1)}{6} + \frac{a\cdot(a+1)}{2} \Big) \\
&= \frac{1}{a^2} \cdot \frac{a\cdot(a+1)\cdot(2a+1) + 3\cdot a\cdot(a+1)}{6} \\
&= \frac{a\cdot(a+1)\cdot(2a+4)}{6\cdot a^2}
\end{aligned}
$$

Therefore, relation (2) becomes:

$$MD(G_{star}) = \frac{(a+1)(a+2)}{3a} \cdot P(l_2 > l_1) + n' \cdot P(l_2 \leq l_1) \qquad (3)$$

It holds that:

$$\begin{aligned} P(l_2 \leq l_1) &= \sum_{i=1}^{a} P(l_2 \leq i) \cdot P(l_1 = i) \\ &= \sum_{i=1}^{a} \frac{i}{a} \cdot \frac{1}{a} \\ &= \frac{1}{a^2} \sum_{i=1}^{a} i \\ &= \frac{a+1}{2a} \end{aligned}$$

Therefore, it is:

$$\begin{aligned} P(l_2 > l_1) &= 1 - P(l_2 \leq l_1) \\ &= \frac{a-1}{2a} \end{aligned}$$

Relation (3) now becomes:

$$MD(G_{star}) = \frac{(a+1)(a+2)}{3a} \cdot \frac{a-1}{2a} + n' \cdot \frac{a+1}{2a}$$

Eventually, the star graph's maximum temporal distance is:

$$MD(G_{star}) = \frac{(a-1)(a+1)(a+2)}{6a^2} + n' \cdot \frac{a+1}{2a}$$

*4.1.2. Case:* $G = K_n$

We will now study extensively the clique's case. First, let us observe that $\delta'(s,t) \leq a$, and therefore $\delta(s,t) \leq a$, for any two vertices $s, t$ in a clique. Hence:

$$MD(K_n) = max_{s,t \in V(K_n)} E\big(\delta(s,t)\big) \leq a$$

*Normalized uniform random temporal clique.* Let $G = K_n$ be a clique of $n$ vertices and let us consider its normalized U-version. That is, every edge $e \in E(K_n)$ is given a single availability label and those labels are chosen randomly and independently from one another from the set $L_0 = \{1, 2, \ldots, n\}$, with the probability that an edge's label equals $i$ being equal to $\frac{1}{n}$, $\forall i \in L_0$.

For any two vertices $s, t$ in the clique, we have:

$$E\big(l(e = \{s,t\})\big) = \frac{n}{2}$$

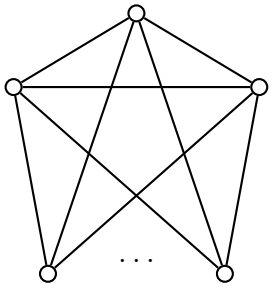


Figure 6: A clique

In the specific case of the normalized uniform random temporal clique of $n$ vertices, there is actually no need for us to assume any *slow* journey to connect any pair of vertices since we already have such a journey, with arrival time equal to $E\big(l(e = \{s,t\})\big) = \frac{n}{2}$. But, for the sake of consistency, we can set the fixed number $n'$ to be equal to $\frac{n}{2}$.

It holds that:

$$MD(normalized\ K_n) = max_{s,t \in V(K_n)} E\big(\delta(s,t)\big) \leq \frac{n}{2}$$

Since this is only an upper bound, we wonder if we can find temporal paths with smaller arrival time than that bound. Indeed, we give a *simple (greedy)* algorithm which can, with high probability, find a journey with small expected arrival time from a given source vertex $s$ to a given target vertex $t$ in the normalized uniform random temporal clique.

*Note.* From here on, the notation "log" will denote the natural logarithm.

**Algorithm 1** The normalized U-RTG clique short journey finding algorithm, Extend-Try

1: **procedure** EXTEND-TRY(*clique* $K_n$, $s$, $t$, $c_1$, $k$)
2: **for** i $= 0$ ... $c_1\sqrt{n}\log n$ **do**
3: $s_i :=$ undefined;
4: **end for**
5: $s_0 := s$;
6: **for** i $= 0$ ... $c_1\sqrt{n}\log n$ **do**
7: **if** $l(\{s_i, t\}) \in \left(c_1\sqrt{n}(\log n)k, c_1\sqrt{n}(\log n)k + \sqrt{n}\right)$ **then**
8: Follow directly the edge $\{s_i, t\}$; Success!
9: go to line 20
10: **else**
11: **if** $\exists u \in U \setminus \{t\}$ (where U stands for the set of the unvisited vertices) such, that $l(\{s_i, u\}) \in \left(k \cdot i, k(i+1)\right)$ **then**
12: $s_{i+1} = u$;
13: go to line 6
14: **else**
15: follow directly the edge $\{s_i, u\}$ with the smallest $l(\{s_i, u\})$ among all $u \in U$; Failure!
16: go to line 23
17: **end if**
18: **end if**
19: **end for**
20: **for** i $= 0$ ... $c_1\sqrt{n}\log n$ **do**
21: **return** $s_i$;
22: **end for**
23: **end procedure**

*Analysis of Extend-Try.* Next, we analyze algorithm 1, looking for the probability that it succeeds.

The probability that the time label of the edge $\{s_i, t\}$ belongs to the interval $(c_1\sqrt{n}k, c_1\sqrt{n}k + \sqrt{n})$ and thus the algorithm succeeds in the $(i+1)^{\text{th}}$ iteration, is:

$$P\Big(l(\{s_i, t\}) \in \left(c_1\sqrt{n}(\log n)k, c_1\sqrt{n}(\log n)k + \sqrt{n}\right)\Big) = \frac{\sqrt{n}}{n} = \frac{1}{\sqrt{n}}$$

Let $\varepsilon_1^j$ be the following event:

"The algorithm finds a proper journey $s_0s_1, s_1s_2, s_2, s_3, \ldots, s_{j-1}s_j$"

meaning that it finds a temporal path, on the temporal edges of which we find strictly ascending time labels and in fact the $i^{th}$ temporal edge's time label correctly belongs to the interval $((i-1)k, ik)$. The time labels are given to the edges independently from one another, thus the probability that the event $\varepsilon_1^j$ occurs is the product of the following probabilities: $P\Big(\exists s_1$ unvisited vertex : the edge $\{s_0, s_1\}$ has time label $l(\{s_0, s_1\}) \in (0, k)\Big)$

$P\Big(\exists s_2$ unvisited vertex : the edge $\{s_1, s_2\}$ has time label $l(\{s_1, s_2\}) \in (k, 2k)$

$\vdots$

$P\Big(\exists s_j$ unvisited vertex : the edge $\{s_{j-1}, s_j\}$ has time label $l(\{s_{j-1}, s_j\}) \in \big((j-1)k, jk\big)$

For any $i^{th}$ probability of the above, it holds that:

$$
\begin{aligned}
&P\Big(\exists s_i \text{ unvisited vertex} : \text{ the edge } \{s_{i-1}, s_i\} \text{ has } l(\{s_{i-1}, s_i\}) \in ((i-1)k, ik)\Big) \\
&= 1 - P\Big(\nexists s_i \text{ unvisited vertex} : \text{ the edge } \{s_{i-1}, s_i\} \text{ has } l(\{s_{i-1}, s_i\}) \in ((i-1)k, ik)\Big) \\
&= 1 - P\Big(\forall s_i \text{ unvisited vertices} : \text{ the edge } \{s_{i-1}, s_i\} \text{ has } l(\{s_{i-1}, s_i\}) \notin ((i-1)k, ik)\Big) \\
&= 1 - \Big(P\big(\text{the edge } \{s_{i-1}, s_i\} \text{ has } l(\{s_{i-1}, s_i\}) \notin ((i-1)k, ik), s_i \text{ unvisited vertex}\big)\Big)^{n-i} \\
&= 1 - \Big(1 - P\big(\text{the edge } \{s_{i-1}, s_i\} \text{ has } l(\{s_{i-1}, s_i\}) \in ((i-1)k, ik), s_i \text{ unvisited vertex}\big)\Big)^{n-i} \\
&= 1 - \Big(1 - \frac{k}{n}\Big)^{n-i}
\end{aligned}
$$

Therefore, the probability that $\varepsilon_1^j$ occurs, is:

$$
\begin{aligned}
P(\varepsilon_1^j) = &\left(1 - \left(1 - \frac{k}{n}\right)^{n-1}\right) \cdot \\
&\left(1 - \left(1 - \frac{k}{n}\right)^{n-2}\right) \cdot \ldots \cdot \left(1 - \left(1 - \frac{k}{n}\right)^{n-j}\right) \geq
\end{aligned}
$$

$$\geq \left(1-\left(1-\frac{k}{n}\right)^{n-j}\right)^{j} \geq$$
$$\geq \left(1-e^{-k}\left(1-\frac{k}{n}\right)^{-j}\right)^{j}$$

For $j \leq c_1\sqrt{n}\log n$, we have:

$$\left(1-\frac{k}{n}\right)^{-j} \leq \left(1-\frac{k}{n}\right)^{-c_1\sqrt{n}\log n} \Leftrightarrow$$
$$\Leftrightarrow 1-e^{-k}\left(1-\frac{k}{n}\right)^{-j} \geq 1-e^{-k}\left(1-\frac{k}{n}\right)^{-c_1\sqrt{n}\log n}$$

and:

$$\left(1-e^{-k}\left(1-\frac{k}{n}\right)^{-j}\right)^{j} \geq \left(1-e^{-k}\left(1-\frac{k}{n}\right)^{-c_1\sqrt{n}\log n}\right)^{c_1\sqrt{n}\log n}$$

As a result, for $j \leq c_1\sqrt{n}\log n$, it is:

$$P(\varepsilon_1^j) \geq \left(1-e^{-k}\left(1-\frac{k}{n}\right)^{-c_1\sqrt{n}\log n}\right)^{c_1\sqrt{n}\log n}$$
$$P(\varepsilon_1^j) \geq 1-e^{-k}\left(1-\frac{k}{n}\right)^{-c_1\sqrt{n}\log n}$$

It holds asymptotically:

$$c_1\sqrt{n}\log n \leq n \Leftrightarrow$$
$$\left(1-\frac{k}{n}\right)^{c_1\sqrt{n}\log n} \geq \left(1-\frac{k}{n}\right)^{n} \Leftrightarrow$$
$$\left(1-\frac{k}{n}\right)^{-c_1\sqrt{n}\log n} \leq \left(1-\frac{k}{n}\right)^{-n} \Leftrightarrow$$
$$1-e^{-k}\left(1-\frac{k}{n}\right)^{-c_1\sqrt{n}\log n} \geq 1-e^{-k}\left(1-\frac{k}{n}\right)^{-n}$$

Therefore:

$$P(\varepsilon_1^j) \geq 1-e^{-k}\left(1-\frac{k}{n}\right)^{-n}$$

and since $k \geq 1$, we have:

$$P(\varepsilon_1^j) \geq 1-e^{-k}\left(1-\frac{1}{n}\right)^{-n}$$

$$= 1 - e^{-k}e$$
$$= 1 - e^{1-k}$$

For $k = r \log n,\ r > 1$, we have:

$$P(\varepsilon_1^j) \geq 1 - e^{1-r\log n}$$
$$= 1 - en^{-r}$$

The probability that we fail in every iteration $i = 0, \ldots, c_1\sqrt{n}\log n$ to find a vertex $s_i$ such, that $l(\{s_i, t\}) \in (c_1\sqrt{n}k, c_1\sqrt{n}k + \sqrt{n})$ is:

$$P(allfail) = \overbrace{\Big(1 - \frac{1}{\sqrt{n}}\Big) \cdot \Big(1 - \frac{1}{\sqrt{n}}\Big) \cdot \ldots \Big(1 - \frac{1}{\sqrt{n}}\Big)}^{c_1\sqrt{n}\log n \text{ factors}}$$
$$= \Big(1 - \frac{1}{\sqrt{n}}\Big)^{c_1\sqrt{n}\log n}$$
$$= e^{-c_1 \log n} = n^{-c_1}$$

The probability that we succeed in some iteration of the algorithm is:

$$P(success) = \Big(1 - P(allfail)\Big)P(\varepsilon_1^j)$$
$$\geq \Big(1 - n^{-c_1}\Big)\Big(1 - en^{-r}\Big)$$

Therefore, the following theorem holds:

**Theorem 1.** *For any constants $c_1, r > 1$, given two vertices $s, t$, $s \neq t$, of the normalized uniform random temporal clique, $K_n$, the probability to arrive, starting from $s$, to $t$ at time at most*

$$t_0 = c_1\sqrt{n}(\log n)k + \sqrt{n},\ \textit{where}\ k = r\log n$$

*is at least*

$$\Big(1 - n^{-c_1}\Big)\Big(1 - en^{-r}\Big).$$

*Remark.* For the "on-line" case, where a traveller starts from $s$ and wants to find a small journey to $t$, but he can only see the edges (*arcs*) out of visited vertices, we conjecture that Algorithm 1 gives a very tight bound on the expected arrival time.

## 5. Temporal Diameter

In this section, we study the concept of the temporal diameter of a uniform random temporal graph.

**Definition 6.** *Consider an instance $G(L)$ of a U-RTG. We denote the maximum of all distributional temporal distances between all pairs of vertices of $G(L)$ by $d(G(L))$:*

$$d(G(L)) = max_{s,t \in V(G)} \delta'(s,t).$$

*We define $diam(G(L)) = min\{d(G(L)), n'\}$. Then, the Expected or Temporal Diameter of $G$, denoted by $TD$, is given by the following formula:*

$$TD(G) = E\Big(diam\big(G(L)\big)\Big) = \sum_{L} diam\big(G(L)\big) \cdot P(L)$$

*, where $P(L)$ is the probability for labelling $L$ to occur.*

We can easily prove that every temporal graph's temporal diameter, $TD$, is equal or greater than its maximum expected temporal distance, $MD$.

**Theorem 2.** *It holds that:*

$$TD(G) \geq MD(G), \text{ for every temporal graph } G.$$

*Proof.* To prove this, we use the Reverse Fatou's Lemma[D10]:

**Theorem** (Reverse Fatou's lemma)**.** *If $X_n \geq 0$, for all $n$, then*

$$E(lim_n sup X_n) \geq lim_n sup E(X_n).$$

In other words, the expected value of the maximum of a set of random variables is at least equal to the maximum of the expected values of those variables.

Now, notice that the Temporal Diameter of a temporal graph $G$ is actually the expected value of the maximum of all distributional temporal distances, that is $E(max_{s,t \in V(G)} \delta'(s,t))$, in the case where we have $\delta'(s,t) \leq n'$, for every pair of vertices $s, t \in V(G)$. In that case, the Maximum Expected Temporal Distance of $G$ is actually the maximum of the expected values of all pairs of vertices' distributional temporal distances, that is $max_{s,t \in V(G)} E(\delta'(s,t))$.

Therefore, in that case, using the above described Reverse Fatou's Lemma, we conclude that:

$$TD(G) \geq MD(G).$$

In the case, where there is at least one pair of vertices $s, t \in V(G)$ such, that $\delta'(s,t) \geq n'$, both the temporal diameter and the maximum expected temporal distance of $G$ are equal to $n'$.

Thus, we conclude that it generally applies that:

$$TD(G) \geq MD(G), \text{ for every temporal graph } G.$$

□

We will now prove that the time $t_0 - o(t_0)$ (see. Theorem 1) is an upper bound of the normalized uniform random temporal clique's temporal diameter, $TD$, and, thus, is an upper bound of its maximum expected temporal distance, $MD$.

**Theorem 3.** *The quantity $t_0 - o(t_0)$ is an upper bound of both the temporal diameter, $TD$, and the maximum expected temporal distance, $MD$, of the normalized U-RT clique.*

*Proof.* Let $s, t$ be two vertices of the normalized U-RT clique. We call $E_{st}$ the following event:

"We arrive, starting from $s$, to $t$ at time at most $t_o$"

where $t_0 = c_1\sqrt{n}(\log n)k + \sqrt{n},\ c_1 > 1, k = r\log n,\ r > 1$.
It holds that:

$$\begin{aligned} P(E_{st}) &\geq \left(1 - n^{-c_1}\right)\left(1 - en^{-r}\right) \\ &\geq 1 - n^{-c_1} - en^{-r} \end{aligned}$$

For $r = c_1$, the above relation becomes:

$$\begin{aligned} P(E_{st}) &\geq 1 - n^{-c_1} - en^{-c_1} \\ &\geq 1 - 2en^{-c_1} \end{aligned}$$

Therefore, the probability that the complement of $E_{st}$ occurs is:

$$\begin{aligned} P(\overline{E}_{st}) &= 1 - P(E_{st}) \\ &\leq 2en^{-c_1} \end{aligned}$$

Thus, the probability that there exist two vertices $s, t$ such that we arrive, starting from $s$, to $t$ at time greater than $t_0$ is:

$$\begin{aligned} P(\exists s, t : \overline{E}_{st}) &\leq n(n-1)2en^{-c_1} \\ &\leq 2en^{-c_1-2} \end{aligned}$$

Let us denote by $T$ the $max\{a_{st}, s, t \in V(K_n)\}$, where $a_{st}$ is the greatest arrival time amongst all $(s,t)$-journeys' arrival times. Then, we have:

$$\begin{aligned} P(\exists s, t : \overline{E}_{st}) &= P(T > t_0) \\ &\leq 2en^{-c_1-2} \end{aligned}$$

It is:

$$\begin{aligned} TD &\leq E(max\{a_{st}, s, t \in V(K_n)\}) \\ &\leq (1 - 2en^{-c_1-2}) \cdot t_0 + n \cdot 2en^{-c_1-2} \\ &\leq t_0 - o(t_0) \end{aligned}$$

Since $TD(G) \geq MD(G)$, for every temporal graph $G$, we conclude that in the case of the normalized U-RT clique, it is:

$$MD \leq TD \leq t_0 - o(t_0)$$

□

## 6. An optimization problem: The Bridges' problem

We will now study an optimization problem concerning the temporal multigraph shown in Figure 7.

***The problem***

$n$ people are located on one bank of a river (see vertex $s$, Figure 7) and want to go to the other side (see vertex $t$, Figure 7). Each one can go across one

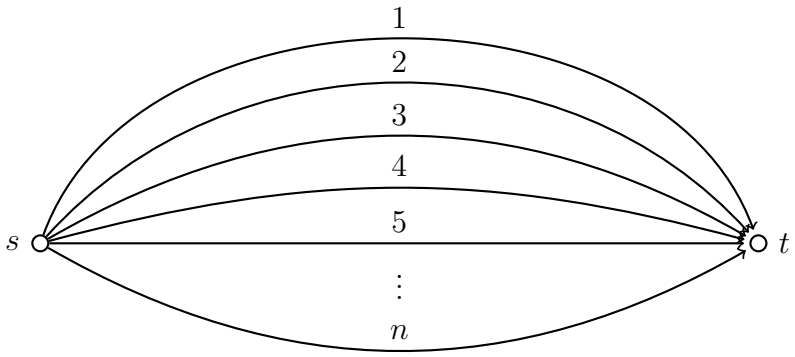



Figure 7: The bridges' problem

of a total of $n$ bridges that connect the two riversides, paying individual cost equal to $1 + \frac{i}{m_i}$, where $i$ stands for the number of the bridge they pass and $m_i$ stands for the total sum of people that cross that bridge. Thus, the total cost payed by $m$ people to cross the $i^{th}$ bridge, $i = 1, 2, \ldots, n$, is:

$$cost[i] = m_i + i$$

We denote by *maximum cost payed* the maximum, over all bridges $i$, cost $m_i + i$:

$$maximum\ cost\ payed = max\{m_i + i, i = 1, 2, \ldots, n :\ bridge\}$$

How should the $n$ people be assigned to the bridges so, that the maximum cost payed is minimized?
We denote the minimum, over all assignments of $n$ people to $n$ bridges, maximum cost payed by $OPT$, that is:

$$OPT = min_{all\ assignments}\{maximum\ cost\ payed\}$$

*Remark.* In another interpretation of the bridges' problem, as we call the above described problem, we consider the multi-labeled temporal digraph of two vertices $s, t$ and one single edge $\{s, t\}$ which is assigned the discrete time labels $1, 2, \ldots, n$ (see figure 8).

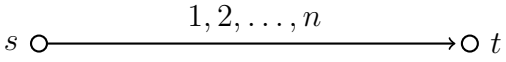



Figure 8: The bridges' problem (another interpretation)

Here we have a single bridge which is available everyday from day 1 to day $n$. As time progresses the cost someone needs to pay to move from $s$ to $t$ increases. Again, one has to pay individual cost equal to $1 + \frac{i}{m_i}$, where $i$ stands for the day on which he decides to move from $s$ to $t$ and $m_i$ stands for the total sum of people decide to move from $s$ to$t$ on that same day. Therefore, the total cost payed by $m$ people who move from $s$ to $t$ on the $i^{th}$ day, $i = 1, 2, \ldots, n$, is:

$$cost[i] = m_i + i$$

**Theorem 4.** *We can compute the assignment of n persons to n bridges that achieves the $OPT$ in polynomial time $O(n^2)$.*

*Proof.* We provide Algorithm 2 and show that it computes the assignment that achieves $OPT$.

**Algorithm 2** The bridges problem solving algorithm

```
1: procedure BRIDGES(n)
2:     cost[] is a 1×n array which holds the bridges' costs;
3:     content[] is a 1×n array which holds the bridges' contents;   ▷ a.k.a
4:                                                                    how
5:                                                                  many people
6:                                                                  are on each
7:                                                                  bridge
8:     m := n;                                          ▷ m is the number of bridges
9:     for i = 1 ... m do
10:        content[i] := 0;                                  ▷ Initializations
11:        cost[i] := i;
12:    end for
```

**Algorithm 2** The bridges problem solving algorithm (continued)

```
13:     for i = 1 ... n do
14:         bridge := 1;    ▷ Initialize the bridge that the i^th person will pass
15:         for  j = 2 ... m do      ▷ Find the bridge that gives the minimum
16:                                                              possible cost
17:             if  cost[j] < cost[bridge]  then
18:                 bridge := j;
19:             end if
20:         end for
21:         content[bridge] := content[bridge]+1;       ▷ Add the i^th person to
22:                                                                the selected
23:                                                            bridge's content
24:         cost[bridge] := cost[bridge]+1;       ▷ Calculate the right new cost
25:     end for
26:     for i = 1 ... m do
27:         if content[i] == 0 then
28:             cost[i] := 0;
29:         end if
30:         if content[i] == 1 then
31:             Write content[i] , " person passes bridge #", i , " who
                therefore has to pay cost equal to ", cost[i];
32:         else
33:             Write content[i] , " people pass bridge #", i , " who therefore
                 have to pay cost equal to ", cost[i];
34:         end if                                    ▷ Print the bridges' costs
35:     end for
36: end procedure
```

The algorithm assigns the $i^{th}$ person to the bridge, for which the current minimum cost is payed. If there are more than one such bridges, the algorithm assigns the $i^{th}$ person to the first one in order. It is trivial to see that the algorithm's running time is $O(n^2)$.

**Proof of correctness** We will prove the validity of the algorithm 2 by induction on the number $n$ of persons.

- For $n = 1$, the algorithm sets the number of bridges to be $m = 1$ and the sole bridge's content and cost to be equal to 1. In the main loop,

the sole person is assigned to the bridge, paying cost equal to:

$$cost[1] = 2$$

So, actually, the algorithm solves the problem for $n = 1$ person.

- Assume that the algorithm solves the problem for $n = k$ people.
- We will show that the algorithm solves the problem for $n = k + 1$ people.

  Before continuing, let us consider the following: Let $n_1, n_2 \in \mathbb{N}$ numbers of people, with $n_1 > n_2$. It is obvious that the minimum possible maximum cost for $n = n_1$ people is at least equal to the minimum possible maximum cost for $n = n_2$ people.

  Let us observe now that the procedures performed by the algorithm in the main loop for $k$ people, and the results obtained through these, are identical to those performed and obtained respectively for $k + 1$ people, except that for $k + 1$ people, there is a $(k + 1)^{th}$ bridge, which throughout the execution of these processes has zero content, and there is also an additional execution of the loop. At the beginning of this $(k+1)^{th}$ execution, the algorithm has already assigned the $k$ people to the brisges in a way that we obtain the minimum possible maximum cost.

  The algorithm, by construction, assigns the people to the bridges in a way that their costs are ordered by (not necessarily strictly) descending order and indeed one of the following two possible events occur:

  $$\begin{cases} \text{all the bridges have the same cost, denoted by } OPT \\ \text{or} \\ \text{some bridges have cost } OPT \text{ and some others have cost } OPT - 1. \end{cases}$$

  In the second case, the algorithm is obviously going to assign the $(k + 1)^{th}$ person to the first in order bridge that has cost equal to $OPT - 1$, thereby maintaining the maximum cost that occurs on the bridges to a minimum, that is $OPT$.

  In the first case, if $r \leq k + 1$ is the number of the last bridge that has positive content, $content[r]$, then it is:

$$\left\{ \begin{array}{l} r + content[r] = OPT \\ \text{But: } content[r] \geq 1 \text{ and so: } r + content[r] \geq r + 1 \end{array} \right\} \Rightarrow OPT \geq r+1$$

Also, since the$(r+1)^{th}$ bridge has zero content, it is:

$$cost[r+1] = r+1$$

The algorithm checks which of the $k+1$ bridges has the minimum cost to assign the $(k+1)^{th}$ person to that bridge. If $OPT = r+1$, then the algorithm assigns the last person to the $1^{st}$ bridge. Otherwise, it assigns it to the $(r+1)^{th}$ bridge. This way, it ensures the minimum possible maximum cost for the $k+1$ bridges.

Therefore, the algorithm solves the problem for $n = k+1$ people.

□

We will now calculate the value of the $OPT$. Again, let us denote by $r$ the number of bridges that have a positive content, i.e. are not empty, in the optimal case which the Algorithm 2 computes. For the sake of brevity, let us also denote by $l_i$ the content of the $i^{th}$ bridge. Since the average cost of the non empty bridges is equal or less than the maximum cost that occurs on those bridges, the following holds for the optimal case:

$$\frac{\sum_{i=1}^{r}(i+l_i)}{r} \leq OPT$$

Therefore, we have:

$$\sum_{i=1}^{r}(i+l_i) \leq rOPT \tag{4}$$

Furthermore, it is easy to see that since, in the optimal case that the algorithm computes, the $OPT$ is greater than any bridge's cost by at most *one*, it holds that:

$$rOPT - r \leq \sum_{i=1}^{r}(i+l_i) \tag{5}$$

By the relations (4) and (5), we have:

$$\begin{aligned}
rOPT - r \leq & \sum_{i=1}^{r}(i + l_i) \leq rOPT \Leftrightarrow \\
rOPT - r \leq & \sum_{i=1}^{r} i + \sum_{i=1}^{r} l_i \leq rOPT \Leftrightarrow \\
rOPT - r \leq & \frac{r(r+1)}{2} + n \leq rOPT \Leftrightarrow \\
OPT - 1 \leq & \frac{(r+1)}{2} + \frac{n}{r} \leq OPT
\end{aligned}$$

Now, the quantity $\frac{(r+1)}{2} + \frac{n}{r}$ is minimized at $r = \sqrt{2n}$ and at that point, its value is equal to $\sqrt{2n} + \frac{1}{2}$. Therefore, we conclude that:

$$OPT = \lceil \sqrt{2n} + \frac{1}{2} \rceil$$

## 7. Conclusions and further research

There are several open problems related to the findings of the present work. We initiated here the random availability of edges where the selection of time-labels, and thus the selection of moments in time at which the edges are available, follows the uniform distribution. There are still other interesting approaches concerning what distribution the selection of time-labels could follow (see F-CASE in Section 2.1). Another approach that is yet to be examined is that of the multi-labeled temporal graphs, on which we could search for statistical properties respective to the ones we studied within the present work. Yet another interesting direction which we did not consider in this work is to find upper bounds on the maximum expected temporal distance and the temporal diameter of any U-RTG (or F-RTG). Further research could also focus on calculating the actual value of these properties, e.g. in the case of the normalized uniform random temporal clique.

## References


[MMCS13] George Mertzios, Othon Michail, Ioannis Chatzigiannakis, and Paul G. Spirakis (2013). *Temporal Network Optimization Subject to Connectivity Constraints* Springer

[KKK00] D. Kempe, J. Kleinberg, and A. Kumar (2000). *Connectivity and inference problems for temporal networks* In Proceedings of the 32nd annual ACM symposium on Theory of computing (STOC)

[MR02] M. Molloy and B. Reed (2002). *Graph colouring and the probabilistic method, volume 23* Springer

[D10] Durrett, R. (2010). *Probability: Theory and Examples, 4th Edition*, Cambridge University Press